OMAP3530/25/15/03 Applications Processor
Silicon Revisions 3.0, 2.1, and 2.0

Silicon Errata

Literature Number: SPRZ278B
February 2008–Revised August 2008

Contents

1 Introduction 4

1.1 OMAP35x Device and Development Support Tool Nomenclature 4

1.2 Revision Identification 5

2 Silicon Revision 3.0 Usage Notes and Known Design Exceptions to Functional Specifications 7

2.1 Usage Notes for Silicon Revision 3.0 7

2.1.1 Performance Limitation On LCD Read/Write Access Through RFBI L4 Port 7

2.1.2 IVA2: IDLE Instruction Must be Executed from L1P or L2 SRAM Under Hardware Emulation (OMAP3530/25 only) 7

2.1.3 Domain Woken Up by Wake-up Dependency Cannot Transition to Inactive State 8

2.1.4 VENC: Last Data Line Missing in PAL 8

2.1.5 Observability Signals Not Functional in OFF Mode 8

2.1.6 Constraints on Module Clocks When Using DVFS 8

2.1.7 Retention Voltage Not Supported 8

2.1.8 UART: Cannot Acknowledge Idle Requests in Smartidle Mode When Configured for DMA Operations 9

2.1.9 Unexpected RFBI Latency for High Frame Rate 9

2.1.10 Cortex-A8 Errata List 9

2.1.11 Display Controller Subsystem (DSS): Limitations Exist When Generating Horizontal and Vertical Timings 10

2.1.12 Camera ISP: IIR Filters in Auto Focus (AF) Engine Should Only Be Used for Auto-Focus 10

2.1.13 High-Speed USB Host Subsystem: Some Limitations Exist When Connecting to External Devices 10

2.2 Silicon Revision 3.0 Known Design Exceptions to Functional Specifications 14

3 Silicon Revision 2.1 Usage Notes and Known Design Exceptions to Functional Specifications 51

3.1 Usage Notes for Silicon Revision 2.1 51

3.2 Silicon Revision 2.1 Known Design Exceptions to Functional Specifications 52

4 Silicon Revision 2.0 Usage Notes and Known Design Exceptions to Functional Specifications 60

4.1 Usage Notes for Silicon Revision 2.0 60

4.2 Silicon Revision 2.0 Known Design Exceptions to Functional Specifications 60

Appendix 1 Revision History 72

OMAP3530/25/15/03 Applications Processor
Silicon Revision 3.0

1 Introduction

This document describes the known exceptions to the functional specifications for the OMAP3530, OMAP3525, OMAP3515, and OMAP3503 applications processors. [See the *OMAP3530/25 Applications Processor Data Manual* (literature number SPRS507) and the *OMAP3515/03 Applications Processor Data Manual* (literature number SPRS505)].

For additional information, see the latest version of the *OMAP35x Technical Reference Manual* (literature number SPRUF98).

The advisory numbers in the document are not sequential. Some advisory numbers have been moved to the next revision. When items are moved, the remaining advisory numbers are not resequenced.

This document also contains "Usage Notes." Usage Notes highlight and describe particular situations where the device's behavior may not match presumed or documented behavior. This may include behaviors that affect device performance or functional correctness. These notes will be incorporated into future documentation updates for the device (such as the device-specific data sheet), and the behaviors they describe will not be altered in future silicon revisions.

1.1 OMAP35x Device and Development Support Tool Nomenclature

To designate the stages in the product development cycle, TI assigns prefixes to the part numbers of all OMAP processors and support tools. Each commercial OMAP platform member has one of three prefixes: X, P, or null (no prefix). Texas Instruments recommends two of three possible prefix designators for its support tools: TMDX and TMDS. These prefixes represent evolutionary stages of product development from engineering prototypes (TMDX) through fully qualified production devices/tools (TMDS).

Device development evolutionary flow:

X	Experimental device that is not necessarily representative of the final device's electrical specifications
P	Final silicon die that conforms to the device's electrical specifications but has not completed quality and reliability verification
NULL	Fully-qualified production device

Support tool development evolutionary flow:

TMDX	Development-support product that has not yet completed Texas Instruments internal qualification testing
TMDS	Fully-qualified development-support product

X and P devices and TMDX development-support tools are shipped against the following disclaimer:

"Developmental product is intended for internal evaluation purposes."

TMS devices and TMDS development-support tools have been characterized fully, and the quality and reliability of the device have been demonstrated fully. TI's standard warranty applies.

Predictions show that prototype devices (X or P) have a greater failure rate than the standard production devices. Texas Instruments recommends that these devices not be used in any production system because their expected end-use failure rate still is undefined. Only qualified production devices are to be used.

1.2 Revision Identification

The device revision can be determined by the symbols marked on the top of the package. Figure 1 provides an example of the OMAP35x Applications Processor device markings.

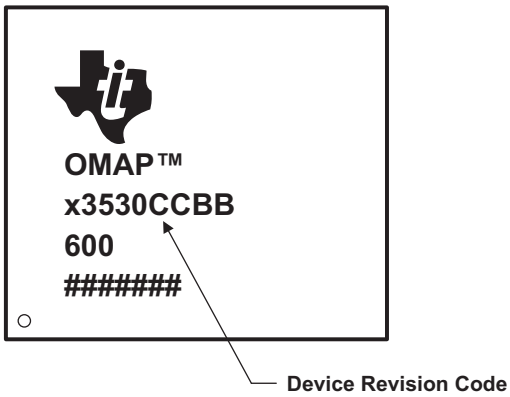


Figure 1. Example, Device Revision Codes for OMAP35x Applications Processor

NOTES:

A. Non-qualified devices are marked with the letters "X" or "P" at the beginning of the device name, while qualified devices have a "blank" at the beginning of the device name.

B. "#" denotes an alphanumeric character.

C. On some "TMX" devices, the device speed may not be shown.

Silicon revision is identified by a code marked on the package. The code is of the format X3530xCBB, where "x" denotes the silicon revision. On TMX devices, if x is"C", then the silicon is revision 3.0. Table 1 and Table 2 list the information associated with each silicon revision on TMX devices.

Table 1. OMAP35x Applications Processor Revision Codes

DEVICE REVISION CODE	SILICON REVISION	COMMENTS
C	3.0	Silicon revision 3.0 (also referred to as ES3.0)
B	2.1	Silicon revision 2.1 (also referred to as ES2.1)

Each silicon revision uses a specific revision of the C64x+ CPU, the C64x+ Megamodule, as well as the ARM Cortex-A8 processor. Table 2 lists the C64x+ CPU and C64x+ Megamodule revision associated with each silicon revision. The C64x+ CPU revision can be read from the REVISION_ID field of the CPU Control Status Register (CSR). The C64x+ Megamodule revision can be read from the REVISION field of the Megamodule Revision ID register (MM_REVID) located at address IVA2.2 subsystem memory address 0181 2000h. The ARM Cortex-A8 variant and revision can be read from the Main ID Register.

The ROM code revision can be read from base address 4001 BFFCh. The ROM code version consists of two decimal numbers: major and minor. The major number is always 14, minor number counts ROM code version. The ROM code version is coded as hexadecimal readable values, e.g. ROM version 14.04 will be coded as 0000 1404h. Table 2 shows the ROM code revision for each silicon revision of the device.

Table 2. Silicon Revision Variables

SILICON REVISION	C64X+ CPU REVISION	C64X+ MEGAMODULE REVISION	ARM CORTEX-A8 VARIANT/REVISION	ROM REVISION
3.0	CPU_ID = 10h REVISION_ID = 02h	MM_REVID[REVISION] = 0h	r2p1	14.04
2.1	CPU_ID = 10h REVISION_ID = 02h	MM_REVID[REVISION] = 0h	r1p1	14.04
2.0	CPU_ID = 10h REVISION_ID = 02h	MM_REVID[REVISION] = 0h	r1p1	14.04

2 Silicon Revision 3.0 Usage Notes and Known Design Exceptions to Functional Specifications

2.1 Usage Notes for Silicon Revision 3.0

Usage Notes highlight and describe particular situations where the device's behavior may not match presumed or documented behavior. This may include behaviors that affect device performance or functional correctness. These notes will be incorporated into future documentation updates for the device (such as the device-specific data manual), and the behaviors they describe will not be altered in future silicon revisions.

Note: The peripherals supported on the various OMAP35x Application Processors are different. The user should only refer to usage notes and advisories pertaining to features supported on the specific device. For a complete list of the supported features of the OMAP35x Application Processors, see the device-specific data manuals.

2.1.1 Performance Limitation On LCD Read/Write Access Through RFBI L4 Port

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, back-to-back accesses for both *Read* and *Write* to the LCD through the L4 interconnect interface of the RFBI module are not supported. A penalty of 1 L4 clock cycle between two consecutive accesses exists.

Read access to the LCD through the RFBI L4 port: The data of a *Read* access is sent back to the initiator of the access only at RECycleTime. RECycleTime is used as a reference event for CS release as well (CS is used as an on-going access notification signal depending on the type of LCD panel connected). This means that any *Read* access to the LCD through the L4 interface of the RFBI will be ended by a CS release (CS going inactive at the end of the access). Therefore, the OMAP35x Applications Processor does not support two consecutive *Read* accesses to the LCD (there is always 1 L4 clock cycle between 2 accesses).

Write access to the LCD through the RFBI L4 port: For a *Write* access, the back-to-back data of a single access that has been split is supported and guaranteed (e.g., one 32-bit write split in two consecutive, back-to-back, 16-bit accesses). In this case, the internal bus CS signal will be kept active until split access completion. However, when there are two different write accesses from the initiator, they will not be seen as back-to-back by the LCD. At the end of the transaction corresponding to one write access from the initiator, the internal bus CS will be released at WECycleTime, and the next command from the initiator will be accepted. Consequently, any *Write* access to the LCD through the L4 interface of the RFBI will be ended by a CS release (CS going inactive at the end of the access). Therefore, the OMAP35x Applications Processor does not support two consecutive data transmits to the LCD (corresponding to two different and consecutive *Write* accesses from the initiator).

(Internal reference number: 2.1)

2.1.2 IVA2: IDLE Instruction Must be Executed from L1P or L2 SRAM Under Hardware Emulation (OMAP3530/25 only)

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, when the C64x core executes an IDLE instruction, and an emulator access to a register or memory happens simultaneously, the CPU may hang, depending on the relative speed of TCK, GEM clock, and the external memory where the IDLE instruction is fetched. This limitation is ***only*** applicable in *emulation* mode when an external emulator, such as TI XDS560, is connected to the device.

To ensure this race condition does not occur, the IDLE instruction should be executed from L1P SRAM of L2SRAM. This will allow the IDLE instruction to execute safely under hardware emulation.

(Internal reference number: 2.4)

2.1.3 Domain Woken Up by Wake-up Dependency Cannot Transition to Inactive State

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, when a domain is activated up by a wake-up dependency, it cannot transition to an inactive (RET/OFF) state until one of the following conditions are satisfied:

1. If the domain has a sleep dependency with another domain, this domain and the sleep dependency must be also activated (hard-wired or enabled by the software).
2. If the domain is an initiator, it must be taken out of *Standby* mode.

Note: The user should ensure that when a wake-up dependency is enabled for a domain, it is expected to be fully active after the wake-up occurs. This issue happens in a case where a domain has been woken-up, but not activated and used.

(Internal reference number: 2.7)

2.1.4 VENC: Last Data Line Missing in PAL

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, the VENC cannot show the image data at the end of the even frames on line 623. This is a minor violation of the ITU-R BT.470-6. Since lines 23 and 336 are reserved for WSS, the VENC can only show 574 lines of data instead of the 576 lines defined in the standard. One half line is missing for lines 23 (reserved for WSS) and 623, and one full line is missing for line 336 (reserved for WSS).

(Internal reference number: 2.8)

2.1.5 Observability Signals Not Functional in OFF Mode

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, signals are routed to observability pins through a buffer powered by VDD2. Consequently, observability is not functional when VDD2 is powered off. This impacts the observability debug feature, but has no functional drawback.

Note: Maintaining the VDD2 supply on the board has no effect since isolation cells are activated regardless of the supplied voltage.

(Internal reference number: 2.10)

2.1.6 Constraints on Module Clocks When Using DVFS

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, module interface timings can depend on the operating point (OPP). The timing closure is done for one instance of a module at a particular voltage. Operating conditions which work on one OPP are not automatically guaranteed on another module's OPP. For more details on the module interface switching characteristics, see the OMAP35x Applications Processor device-specific data manual.

When Dynamic Voltage and Frequency Scaling (DVFS) is performed, the software should ensure that no timing violations will occur by the two following methods:

- Using software, reconfigure the module accordingly when DVFS is performed to avoid timing violations.
- Use a conservative frequency on the module interface clock corresponding to the performance of the lower OPP used to ensure that DVFS can be performed without additional software management.

(Internal reference number: 2.11)

2.1.7 Retention Voltage Not Supported

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, the retention voltage ***is not*** supported. The TI 65nm process defines a retention voltage which is lower than the lowest OPP voltage. This retention voltage is defined assuming that there is no logic activity at all in the device, which does not apply to the OMAP35x Applications Processor where asynchronous signals (wake-up) can be triggered during the device RET state. Because of this system requirement, the retention voltage does not apply to the OMAP35x Applications Processor, and the lowest OPP voltage should be used instead of a specific retention voltage.

The lowest operating point voltage should be used when the OMAP35x Applications Processor is in *retention* mode.

Note: This voltage can be optimized using SmartReflex before the device goes into *retention* mode.

(Internal reference number: 2.14)

2.1.8 UART: Cannot Acknowledge Idle Requests in Smartidle Mode When Configured for DMA Operations

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, when configured for DMA operations using *smartidle* mode (SYSC[4:3].IDLEMODE = 0x2), the UART module will not acknowledge incoming idle requests. As a consequence, it can prevent L4 from going to idle.

When there are additional expected transfers, the UART should be placed in *force-idle* mode.

(Internal reference number: 2.15)

2.1.9 Unexpected RFBI Latency for High Frame Rate

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, the Remote Frame Buffer (RFBI) state machine architecture can create non-optimized pipelining of the RFBI output data stream. As a result, some idle cycles may be inserted between RFBI accesses to the external panels. The number of idle cycles added depends on the OMAP clock configuration and RFBI configuration. Table 3 shows the minimum additional number of cycles depending on the RFBI configurations.

Table 3. Additional Minimum Cycletime (CS/WE Always Asserted)

RFBI PERFORMANCE	RFBI_CONFIG. CYCLEFORMAT	RFBI_CONFIG. OCPFORMAT	MIN CYCLETIME (Number of OCP cycles)
OCP Slave	1 pixel/ cycle	1 pixel	5
	1 pixel/ 2 cycles	1 pixel	4
	1 pixel/ 3 cycles	1 pixel	4
	2 pixels/ 3 cycles	1 pixel	6
	1 pixel/ cycle	2 pixels	4
	1 pixel/ 2 cycles	2 pixels	4
	1 pixel/ 3 cycles	2 pixels	4
	2 pixels/ 3 cycles	2 pixels	6
Display Controller	1 pixel/ cycle	N/A	4
	1 pixel/ 2 cycles	N/A	3
	1 pixel/ 3 cycles	N/A	3
	1 pixel/ 3 cycles	N/A	6

(Internal reference number: 2.20)

2.1.10 Cortex-A8 Errata List

The *OMAP3530/25/15/03 Applications Processor Silicon Revisions 3.0, 2.1, and 2.0 Silicon Errata* does not cover the advisories associated with the Cortex-A8 processor. For a list of the advisories associated with the each version of the Cortex-A8 processor, contact your TI representative for a copy of the *ARM Core Cortex-A8 (AT400/AT401) Errata Notice*. See Table 2 to determine which version of the Cortex-A8 processor is included in each OMAP35x silicon revision.

2.1.11 Display Controller Subsystem (DSS): Limitations Exist When Generating Horizontal and Vertical Timings

The display controller registers that control horizontal and vertical output timings (HBP, HFP, HSW, etc.) are not flexible enough to generate waveforms/timings required by some video display standards such as those of the Video Electronics Standards Association (VESA). Table 4 below describes which register fields are used to control the vertical and horizontal output timings, the current register field widths, and the required register field widths.

Table 4. Display Controller Horizontal and Vertical Timing Control Register Fields

Field Name	DSS Register Field	Field Width (Bits)	Required Width (Bits)
HBP	DISPC_TIMING_H[27:20]	8	12
HFP	DISPC_TIMING_H[15:8]	8	12
HSW	DISPC_TIMING_H[5:0]	6	8
VPB	DISPC_TIMING_V[27:20]	8	12
VFP	DISPC_TIMING_V[15:8]	8	12
VSW	DISPC_TIMING_V[5:0]	6	8

2.1.12 Camera ISP: IIR Filters in Auto Focus (AF) Engine Should Only Be Used for Auto-Focus

The AF engine IIR filters should only be used for auto-focus purposes. Any other use for the IIR filters is not supported.

2.1.13 High-Speed USB Host Subsystem: Some Limitations Exist When Connecting to External Devices

As shown in Figure 2, the OMAP35x device includes a high-speed universal serial bus (USB) OTG controller and a high-speed USB host subsystem.

Note: USB Port 3 is not available on CUS package.

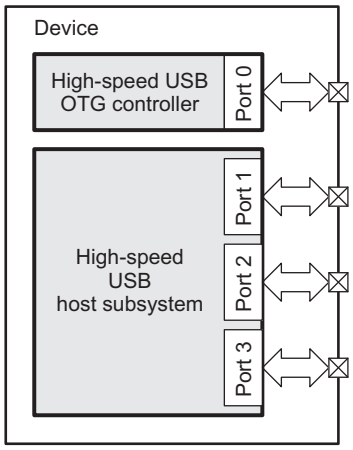


Figure 2. OMAP35x USB Modules

The high-speed USB OTG controller supports a single USB port which uses a UTMI low-pin interface (ULPI) to connect to an off-chip transceiver (12-pin/8-bit single-data rate mode). As shown in Figure 3, USB Port 0 can be connected to the USB 2.0 PHY included in OMAP35x companion chips, e.g. TPS69xxx. Alternatively, USB Port 0 can be connected to discrete USB 2.0 PHYs which include a UTMI low-pin interface (ULPI) and are capable if sourcing an output clock.

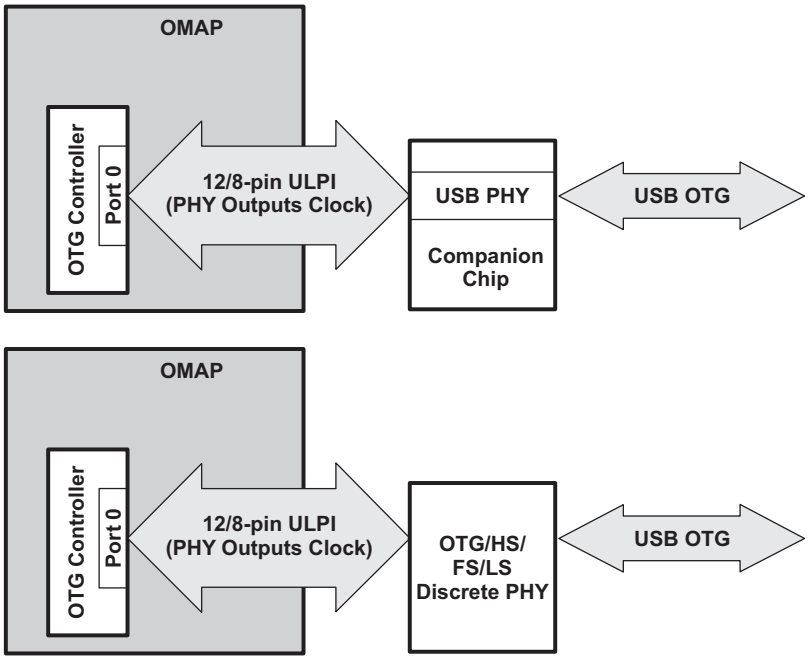


Figure 3. Typical Uses for USB Port 0

As shown in Figure 4, the high-speed universal serial bus (USB) host subsystem supports up to three USB ports, each one of which can owned by one of two controllers:

- The EHCI controller, based on the Enhanced Host Controller Interface (EHCI) specification for USB Release 1.0, is in charge of high-speed traffic (480M bit/s) over either a UTMI port or a UTMI low-pin interface (ULPI) port.
- The OHCI controller, based on the Open Host Controller Interface (OHCI) specification for USB Release 1.0a, is in charge of full-speed/low-speed traffic (12/1.5M bit/s, respectively) over a serial interface.

Both the OHCI and EHCI controllers can operate in parallel. Each of the three USB ports is owned by exactly one of the controllers at any time. Also, when a controller uses the UTMI port, the USB TLL block converts that port to a ULPI transceiver-less link logic (TLL) format.

Please note the following functional limitations when using the high-speed USB host subsystem:

- On silicon revision 3.0 and earlier, if one port is configured in the ULPI mode, then all other ports must use the same configuration. Therefore, the ports must be configured high-speed mode or full-speed/low-speed mode.
- USB Port 3 cannot operate in ULPI mode; it can only operate in serial or ULPI TLL mode. Furthermore, USB Port 3 is not available in the CUS package.

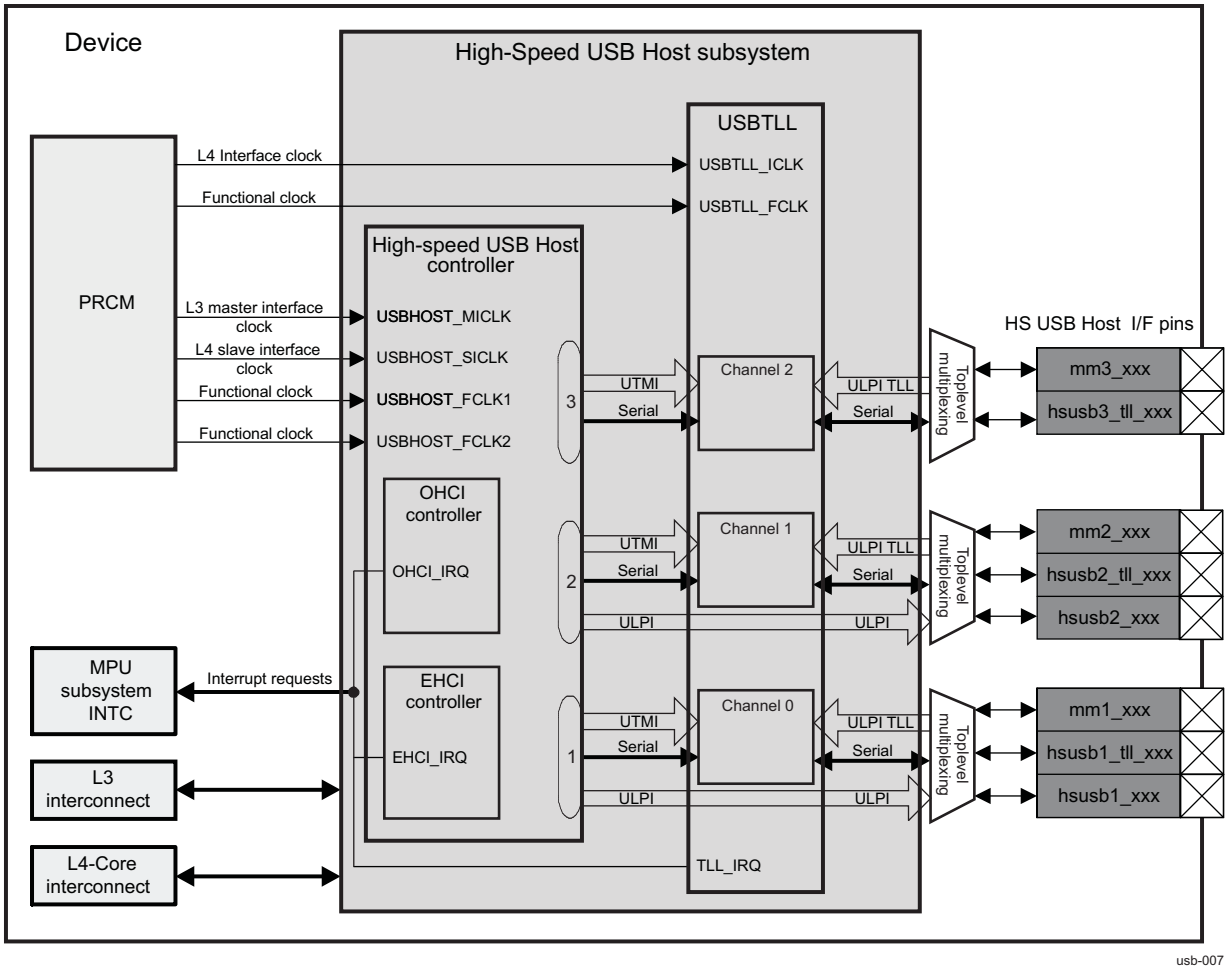
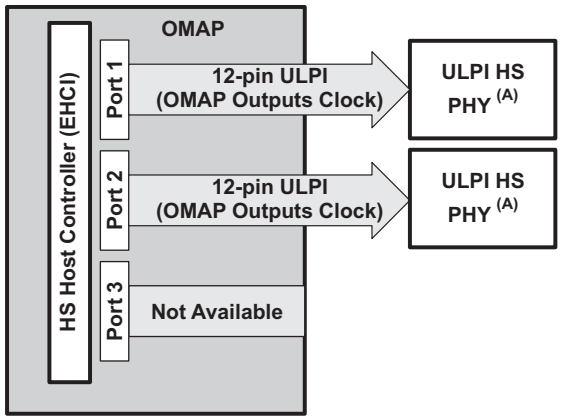


Figure 4. High-Speed USB Host Subsystem Highlight

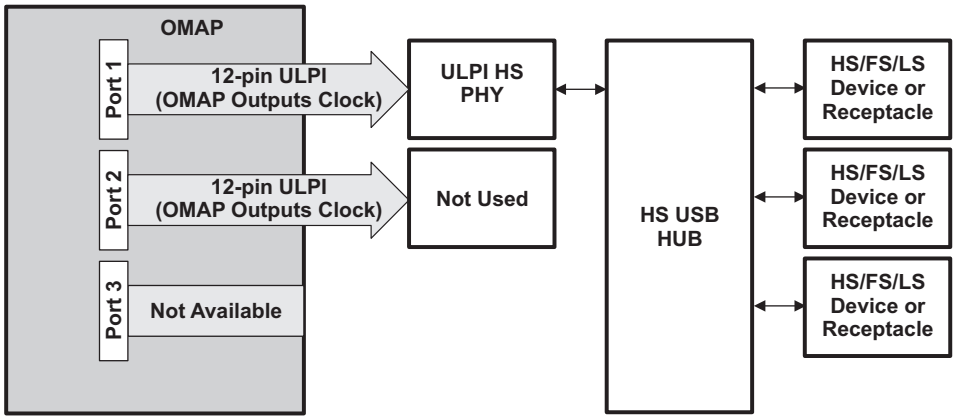
As shown in Figure 5, USB Port 1 and Port 2 can be used to connect to external high-speed PHYs which include a ULPI port. However, in this case, the external USB PHY must be able to accept a source clock generated by the OMAP high-speed USB controller. Also, in this usage model the USB ports cannot support full-speed and low-speed operation. Therefore, using this approach, USB Port 1 and Port 2 cannot provide a fully compliant USB 2.0 Type-A receptacle; a high-speed USB hub would be required in this case; please see Figure 6.

Note: USB Port 3 does not support ULPI mode. USB Port 3 is not available on CUS package.



A. USB port 3 does not support ULPI mode.
Also, USB port 3 not available on CUS package.

Figure 5. Connecting to High-Speed PHYs Using USB Ports



A. USB port 3 does not support ULPI mode. Also, USB port 3 not available on CUS package.

Figure 6. Connecting to a High-Speed USB Hub

2.2 Silicon Revision 3.0 Known Design Exceptions to Functional Specifications

Table 5. Silicon Revision 3.0 Advisory List

Title	Page
Advisory 3.0.1.2 I2C Module Does Not Allow 0-byte Data Requests	15
Advisory 3.0.1.4 Delay Required to Read Some GP, WD, and Sync Timer Registers After Wakeup	16
Advisory 3.0.1.9 L1D Cache : C64x+ L1D Cache May Lose Data or Hang DMA Operations Under Certain Conditions (OMAP3530/25 only)	17
Advisory 3.0.1.11 Race Condition May Cause I2C Slave to Nack a Transfer	18
Advisory 3.0.1.12 MDR1 Access Can Freeze UART Module When in IrDa Mode	19
Advisory 3.0.1.13 IVA2: Block Cache Operations Word Count (*WC) Must be Less Than or Equal to 0xFF80 (OMAP3530/25 only)	20
Advisory 3.0.1.15 I2C: RDR Flag may be Incorrectly Set	21
Advisory 3.0.1.16 IVA2: EDMA Channel Priority is Not Correctly Enforced (OMAP3530/25 only)	22
Advisory 3.0.1.29 Inactive State Management: Impossible to Transition to OFF or RETENTION States	23
Advisory 3.0.1.36 Inappropriate Warm Reset Generation on Smart Reflex I2C Error	24
Advisory 3.0.1.41 CPU: Back-to-Back SPLOOPS With Interrupts can Cause Incorrect Operation on C64x+ CPU (OMAP3530/25 only)	25
Advisory 3.0.1.42 IVA2: DSP Generates False Internal Exception for Multiple Writes (OMAP3530/25 only)	26
Advisory 3.0.1.53 GPMC may Stall After 256 Write Accesses in NAND_DATA, NAND_COMMAND, or NAND_ADDRESS Registers	28
Advisory 3.0.1.57 SPI Dummy DMA RX Request Generation	29
Advisory 3.0.1.66 SDMA: DMA4_IRQSTATUS_Lx and DMA4_IRQENABLE_Lx Registers are not Secure	31
Advisory 3.0.1.74 PRM_VOLTCTRL and PRM_CLKSRC_CTRL Registers Reset on Warm Reset	32
Advisory 3.0.1.75 IVA2: CAM/SGX Dependencies (OMAP3530/25 only)	33
Advisory 3.0.1.76 Erroneous SResp Generation Issued to AES Immediately After Soft Reset	34
Advisory 3.0.1.77 MPU L2 Cache Size Status Register Value Inverted	35
Advisory 3.0.1.81 L3 Interconnect Clock Divisor Default Value Must be Modified Before Configuration of SDRAM Controller	36
Advisory 3.0.1.112 DMA: Drain_IE Reset Value	37
Advisory 3.0.1.113 sDMA: Channel is Not Disabled After a Transaction Error	38
Advisory 3.0.1.117 Unexpected Stalling May Occur During SDMA/IDMA Accesses to DSP L2 Memory	39
Advisory 3.0.1.119 USB DEVICE Aborts Remote Wakeup Sequence When OMAP35x Device Wakes From OFF/RET to ON in USB TLL Mode	43
Advisory 3.0.1.127 Pending EDMA Interrupts in Video Hardware Accelerator Prevent IVA2 From Going Into Idle Mode	44
Advisory 3.0.1.129 TV detect AC coupling Mode Not Supported	45
Advisory 3.0.1.130 USB DMA Cannot Handle Concurrent Channels	46
Advisory 3.0.1.131 VC1 En/De-coded Bit Stream Corrupted When iLF is Used	47
Advisory 3.0.1.136 ISP: LSC Issue When Used Concurrently With Resizer	48
Advisory 3.0.1.137 DALL Leakage When Placed in Low-Power Stop Mode	49
Advisory 3.0.1.140 ISP Lens Shading Correction Issue	50

Advisory 3.0.1.2 *I2C Module Does Not Allow 0-byte Data Requests*

Revision(s) Affected	3.0 and earlier
Details	When configured as the master, the I2C module does not allow 0-byte data transfers. **Note**: Programming I2Ci.I2C_CNT[15:0]: DCOUNT = 0 will cause undefined behavior.
Workaround(s)	No workaround. ***Do not*** use 0-byte data requests.

Advisory 3.0.1.4 *Delay Required to Read Some GP, WD, and Sync Timer Registers After Wakeup*

Revision(s) Affected

3.0 and earlier

Details

If a General Purpose Timer (GPTimer) is in *posted* mode (TSIRC.POSTED = 1), due to internal resynchronizations, any values read in TCRR, TCAR1, and TCAR2 registers immediately after the timer interface clock (L4) goes from a stopped state to an active state may not return the expected values. This situation is most likely when the OMAP35x Applications Processor wakes up from an idle state.

Notes:

- GPTimer non-posted synchronization mode is not impacted by this advisory.
- This advisory also impacts reads from Watchdog timers WCRR registers.
- All of the watchdog timers support only *posted* internal synchronization mode. There is no capability to change the internal synchronization scheme to *non-posted* mode via software.
- The 32K sync timer CR and 32SYNCNT_REV registers are also impacted by this advisory, since the 32K sync timer is always in *posted* synchronization mode.

Workaround(s)

The software must wait at least 2 timer interface clock cycles + 1 timer functional clock cycle after L4 clock-wakeup before reading TCRR, TCAR1, or TCAR2 registers for GP Timers in *posted* internal synchronization mode, and before reading the WCRR register of the Watchdog timers. The same workaround must be applied before reading CR and 32KSYNCNT_REV registers of the synctimer module.

Advisory 3.0.1.9 *L1D Cache : C64x+ L1D Cache May Lose Data or Hang DMA Operations Under Certain Conditions (OMAP3530/25 only)*

Revision(s) Affected

3.0 and earlier

Details

Under certain conditions, parallel loads with predication to the same cache line may cause victims to be dropped and/or DMA to hang.
All of the following conditions ***must*** be true in order for this problem to occur:

1. Two LD instructions in parallel.
2. Both are LDs to the same cache line (upper 26 address bits are the same).
3. The LD using T1 is predicated and the predicate is *false.*
4. The LD using T2 is either not predicated, or is predicated and the predicate is *true.*
5. The cache line is absent from the cache.
6. The two other lines in the same L1D set are valid.
7. The LRU cache line in the set is dirty.

Results:

- L1D informs L2 to expect a victim for the affected set.
- L2 stalls DMAs with addresses that correspond to that set (DMA includes accesses from IDMA and EDMA).

  Note: DMA includes accesses from IDMA, EDMA, and any external masters, such as PCI or other CPUs.

- L1D processes the true-predicated request correctly.
- L1D does not send the indicated victim.

Impact:

- If the load instruction reads a cacheable location:
  - The updated data in the LRU line gets dropped.
  - DMA accesses whose addresses match the affected set hang.
- If the load instruction reads a non-cacheable location:
  - L1D retains the updated data from the LRU line.
  - DMA reads may see stale data if the LRU line's address is in L2 memory.

Workaround(s)

Use Code Gen patch 6.0.3 (available on update advisor) to recompile your source code and avoid this issue. Libraries supplied by TI will be re-released using the 6.0.3 compiler patch. Customer-generated libraries from TI's third-party supplier may also need to be recompiled.

For existing object code and libraries, an available Perl script can determine locations of parallel predicated loads that may fail. The script is available at the same update advisor location as the Code Gen patch.

Advisory 3.0.1.11 *Race Condition May Cause I2C Slave to Nack a Transfer*

Revision(s) Affected 3.0 and earlier

Details If the I2C module is configured as a slave, in *autoidle* mode (I2C_SYSC.AUTOIDLE = 1) and the ARDY (I2C.12C_STAT[2]) condition and the START condition are detected in the module at the same time, internal clock gating will be incorrectly applied. This will cause the I2C to NACK (I2C.12C_STAT[1]) the transfer for which the START (I2C.I2C_STA[6]) condition was received. Subsequent transfers will be ACKed as expected.

Workaround(s) **Workaround 1**: Software ***must*** set SYSC_AUTOIDLE to 0. In this case, the failure condition never occurs.

Workaround 2: Ensure that the external I2C master always resends a NACKed transfer via software. If a transfer was NACKed because of this race condition, the next transfer will always be ACKed.

Advisory 3.0.1.12 *MDR1 Access Can Freeze UART Module When in IrDa Mode*

Revision(s) Affected 3.0 and earlier

Details Because of a glitchy structure inside the UART module, accessing the MDR1 register may create a dummy underrun condition and freeze the UART IrDa transmission. Only IrDa modes *Slow Infrared* (SIR), *Medium Infrared* (MIR), and *Fast Infrared* (FIR) are impacted. Even if the bug condition occurs in *UART* mode or *IrDa CIR* mode, it will have no effect. Therefore, UART1 and UART2 are immune to this problem, and only UART3 may exhibit this issue when used in one of the IrDa modes– SIR, MIR, or FIR.

Workaround(s) To ensure this problem does not occur, the following software initialization sequence must be used each time MDR1 must be changed to one of the three failing *IrDa* modes:

1. If needed, setup the UART by writing the required registers, except MDR1.
2. Set appropriately the MDR1.MODE_SELECT bit field.
3. Wait for 5 L4 clock cycles + 5 UART functional clock cycles.
4. Read RESUME register to resume the halted operation.

Advisory 3.0.1.13 *IVA2: Block Cache Operations Word Count (*WC) Must be Less Than or Equal to 0xFF80 (OMAP3530/25 only)*

Revision(s) Affected 3.0 and earlier

Details When performing any block cache operation, such as "Writeback", "Writeback with Invalidate", or "Invalidate", for any memory controller or memory range (e.g., L1P, L2, L1D) the word count programmed ***must be*** less than or equal to 0xFF80. If a value greater than 0xFF80 is desired, then this must be broken into multiple operations. The following registers are affected: L2WWC, L2WIWC, L2IWC, L1PIWC, L1DWIWC, L1DWWC, and L1DIWC.

Workaround(s) No workaround.

Advisory 3.0.1.15 *I2C: RDR Flag may be Incorrectly Set*

Revision(s) Affected 3.0 and earlier

Details Under certain rare conditions, the I2C_STAT[13].RDR bit may be set as well as the corresponding interrupt fire, even when there is no data in the receive FIFO, or the I2C data transfer is still ongoing. These spurious RDR events must be ignored by the software.

Workaround(s) Software must filter out unexpected RDR pulses, using the flowchart illustrated in Figure 7 when receiving an I2C RDR interrupt.

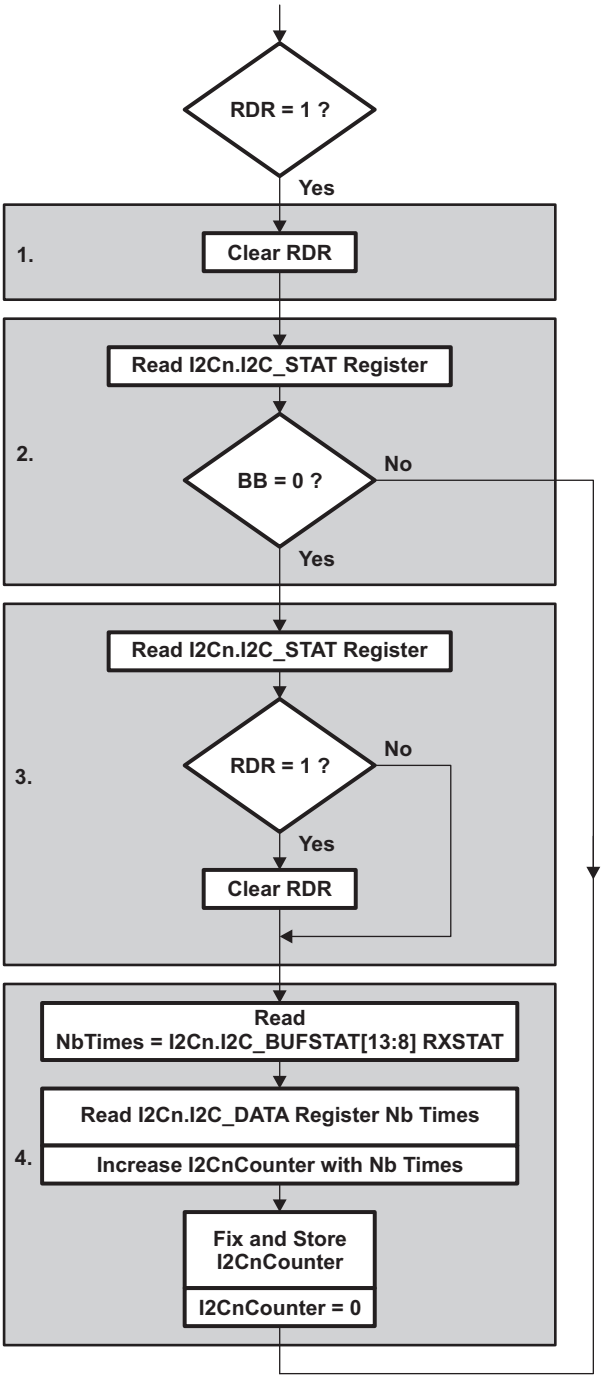


Figure 7. I2C Flowchart

Advisory 3.0.1.16 *IVA2: EDMA Channel Priority is Not Correctly Enforced (OMAP3530/25 only)*

Revision(s) Affected

3.0 and earlier

Details

The EDMA channel controller (TPCC) has 2 event queues: Q0 and Q1. Each queue can be mapped to one of the two Transfer Controllers (TPTC): TPTC0 or TPTC1. As explained in the *OMAP35x Technical Reference Manual* (literature number SPRUF98), the events in each event queue will be extracted as soon as the corresponding TPTC is available for a new Transfer Request (TR) to be programmed into the TPTC. However, due to an issue in the IVA2.2 subsystem, the requests queued in Q1 cannot be submitted to their TPTC as long as Q0 is not empty.

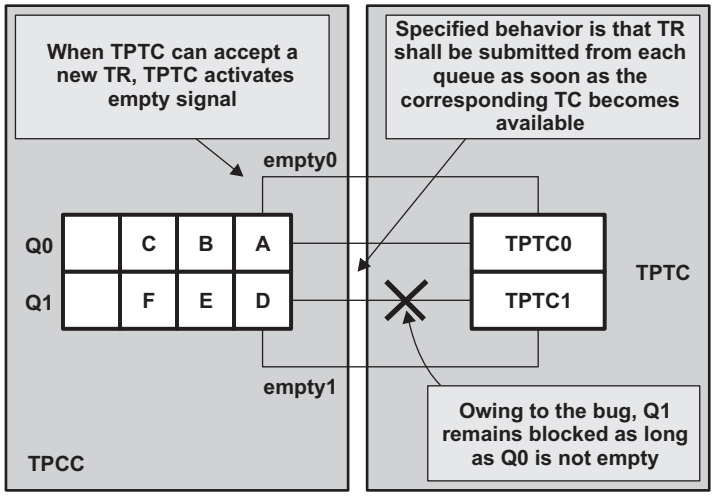


Figure 8. TR Submission Scheme

Workaround(s)

Infrequent short transfers (e.g., latency critical synchronized transfers to/from peripheral) must be placed in Q0, and longer transfers (e.g., block copy) in Q1. For example, when a latency critical transfer is placed in EDMA and a longer transfer in QDMA, the following programming model will minimize the impact of this limitation:

- EDMA Event is queued to Q0 (using IVA_TPCC.DMAQNUMn register, n from 0 to 7)
- QDMA Event is queued to Q1 (using IVA_TPCC.QDMAQNUM register)
- Queue to TC Mapping is Q0:TPTC1 Q1:TPTC0 (using IVA_TPCC.QUETCMAP register)
- Queue Priority is Q0 = 0x0, Q1 = 0x7 (using IVA_TPCC.QUEPRI register)

Advisory 3.0.1.29 *Inactive State Management: Impossible to Transition to OFF or RETENTION States*

Revision(s) Affected 3.0 and earlier

Details If a power domain meets the conditions to go to an INACTIVE state (i.e., POWERSTATE is programmed to 0x3(ON) and clock can be shut off), then the domain will transition to INACTIVE state. However, the domain cannot go to RET or OFF state automatically from INACTIVE state, even if software updates the POWERSTATE bit to 0x1(RET) or 0x0 (OFF). The domain must be transitioned to the ON state before it can transition to the RET or OFF states.

Workaround(s) The following two conditions must be met:

1. Do not use autostate.
2. Perform wake-up event (software must force wakeup) to transition to an active state before transitioning to the RET or OFF states.

Advisory 3.0.1.36 *Inappropriate Warm Reset Generation on Smart Reflex I2C Error*

Revision(s) Affected

3.0 and earlier

Details

The Voltage Controller generates an I2C access error when an I2C command is sent to PowerIC during a voltage domain sleep transition. When the access error is generated, an interrupt is generated for software error handling. PRCM generates a warm_reset when an access error is generated by the Voltage Controller during voltage domain wake-up transition. This is to recover the full system (OMAP35x Applications Processor + OMAP Peripherals + Power IC) and avoid a deadlock (OMAP35x Applications Processor wakeup transition stalling due to I2C access error and VDD1/ VDD2 not supplied). Since both VDD1 and VDD2 Voltage Controllers share the same interrupt line, an access error for VDD2 Voltage controller is also propagated to VDD1 Voltage controller.

This bug occurs only with the I2C module used by the Smart Reflex module. Other instances of the I2C module are not impacted by this issue.

Workaround(s)

In the following scenario,

1. VDD1 is performing a wake-up transition while VDD2 is performing a sleep transition.
2. An I2C access error is generated on VDD2 sleep request and VDD1 and VDD2 share the same error line, so this access error on VDD2 is broadcasted to both VDD1 and VDD2, the condition for a warm_reset generation is met on VDD1, and Warm_reset is asserted inappropriately.

This issue was detected in simulation, but it is a corner case which is not expected to occur in production since the I2C access error should never occur if the PCB is safe.

Advisory 3.0.1.41 *CPU: Back-to-Back SPLOOPS With Interrupts can Cause Incorrect Operation on C64x+ CPU (OMAP3530/25 only)*

Revision(s) Affected

3.0 and earlier

Details

Back-to-back software pipeline loops (SPLOOPs) with interrupts can cause incorrect operation on the C64x+ CPU. This bug occurs when the first SPLOOP is interrupted and there are less than 2 execute packets between the SPKERNEL of the first SPLOOP block (SPKERNEL instruction marks the end of the first SPLOOP block) and the SPLOOP instruction of the second SPLOOP block (SPLOOP instruction marks the beginning of the second SPLOOP block). The first SPLOOP block terminates abruptly (i.e., without completing the loop, even though the termination condition is false). The failure mechanism can be seen as a hang or by the first SPLOOP block draining for the interrupt and starting the second SPLOOP block without taking the interrupt or returning to complete the first SPLOOP block.

Workaround(s)

The C6000 compiler release v6.0.6 and above detects this problem. If there are fewer than 2 execute packets between the SPKERNEL and SPLOOP instructions, the compiler will add the appropriate number of NOP instructions following the SPKERNEL instruction.

For example,

```
...
SPKERNEL     0, 0
NOP          1; SDSCM00012367 HW bug workaround
MVK      .L1 0x1,A0
[A0] SPLOOPW 3;12
NOP          1
...
```

The assembler will detect sequences that could potentially trigger this bug, and issue a remark. For example,

```
"neg_test.asm", REMARK at line 21 [R5001] SDSCM00012367 potentially
triggered by this execute packet sequence. SLOOP must be at
least 2 EPs away from previous SPKERNEL for safe interrupt
behavior.
```

Note: The assembler tool, asm6x.exe, can be used to determine if a previous version of the compiler generated code that could potentially be affected by this silicon issue. The assembler can also be used on assembly source code to see if the source could be affected by this issue. Replace the old version of asm6x.exe with the 6.0.6 asm6x.exe in your current build setup and recompile or reassemble.
(Internal Tracking Number: 4)

Advisory 3.0.1.42 *IVA2: DSP Generates False Internal Exception for Multiple Writes (OMAP3530/25 only)*

Revision(s) Affected

3.0 and earlier

Details

A false internal exception can be generated by C64x+ if an interrupt happens during DSP code flow and the instructions that will be annulled during pipeline flush are dependant. This issue occurs in the exception detection logic. It examines the DSP instructions during the pipeline flush, even though they have been annulled. The hardware does not detect this and incorrectly assumes that multiple write instructions to the same register with 2 different conditional registers will be executed.

The DSP generates an incorrect internal exception in the following scenario: the CPU is draining the pipeline as part of an interrupt context switch. During this time, it annuls instructions in the pipeline. The first annulled execute packet changes the value of one or more predicate registers. The second annulled execute packet has two or more predicated instructions that use the predicates written in the previous cycle. The values held in the predicate registers appear to cause the instructions in the second annulled execute packet to write to the same register. The conflicting writes would not happen if the first execute packet was not annulled.

Example:

```
ZERO A0
ZERO A1
------------> (interrupt occurs here)
MVK 1, A0; (annulled)
[!A0] MVK 2, A1; (annulled)
|| [!B0] MVK 3, A1; (annulled)
```

Invalid exception triggers as it appears that the last two MVK will both write A1.

Even if this issue appears in a DSP code, it does not affect the code flow and does not produce an unexpected exit routine value.

Workaround(s)

The CPU only recognizes the incorrect exception while it drains the pipeline for an interrupt. As a result, the CPU begins exception processing upon reaching the interrupt handler. The NRP (NMI Return Pointer Register) and NTSR (NMI Task State Register) will reflect the state of the machine upon arriving at the interrupt handler.

Therefore, to identify the incorrect resource conflict exception in software, verify the following conditions at the beginning of the exception handler prior to normal exception processing:

1. Exception occurred during an interrupt context switch.
   - In NTSR, verify that INT=1, SPLX=0, IB=0, CXM=00.
   - Verify that NRP points to an interrupt service fetch packet. That is, (NRP & 0xFFFFFE1F) == (ISTP & 0xFFFFFE1F).
2. The exception is a resource conflict exception. In IERR, verify that RCX == 1 and all other IERR bits == 0.
3. The exception is an internal exception. In EFR, verify that IXF == 1 and all other EFR bits == 0.

Upon matching the above conditions, suppress the exception as follows:

- Clear EFR.IXF by writing 2 to ECR.
- Resume the interrupt handler by branching to NRP.

The above workaround identifies and suppresses all cases of the incorrect resource conflict exception. It resumes normal program execution when the incorrect exception occurs, and has minimal impact on the execution time of program code. The interrupted code sequence runs as expected when the interrupt handler returns.

The workaround also suppresses a particular valid exception case that is indistinguishable from the incorrect case. Specifically, the code will suppress the exception generated by two instructions with different delay slots (e.g., LDW and

DOTP2) writing to the same register in the same cycle, where the conflicting writes occur during the interrupt context switch.

Example of sequence with incorrectly suppressed exception:

```
LDW *A0, A1
DOTP2 A3, A2, A1
NOP
-----------------> interrupt occurs
NOP
NOP ; Both LDW and DOTP2 write to A1 this cycle
```

The workaround will not suppress these valid resource conflict exceptions if the multiple writes occur outside an interrupt context switch. That is, the workaround will not suppress the exception generated by the code above when it executes without an interfering interrupt.

Advisory 3.0.1.53 *GPMC may Stall After 256 Write Accesses in NAND_DATA, NAND_COMMAND, or NAND_ADDRESS Registers*

Revision(s) Affected

3.0 and earlier

Details

The GPMC may stall if the following conditions are met:

1. GPMC_CONFIG[0].NANDFORCEPOSTEDWRITE=1.
2. Software performs more than 256 continuous write accesses in NAND_COMMAND_x, NAND_ADDRESS_x or NAND_DATA_x registers.
3. GPMC_STATUS[0].EMPTYWRITEBUFFERSTATUS is always 0 (buffer not empty) during write accesses. This means the software has to write fast enough in GPMC registers in order to never have the write buffer empty.

This mechanism is CS independent. If the software performs 128 write accesses in NAND_DATA_0 followed by 128 write accesses in NAND_DATA_1 then the bug will occur.

Workaround(s)

Accesses performed through the "prefetch and write posting engine" of the GPMC are not impacted by this limitation, and software should use this mechanism for the best performance.

If the prefetch and write posting engine is not used, when GPMC_CONFIG[0].NANDFORCEPOSTEDWRITE=1 and after 255 write accesses in NAND_COMMAND_x, NAND_DATA_x or NAND_ADDRESS_x registers, the software has to wait until GPMC_STATUS[0].EMPTYWRITEBUFFERSTATUS=1 before sending the next 255 write accesses.

Advisory 3.0.1.57 *SPI Dummy DMA RX Request Generation*

Revision(s) Affected 3.0 and earlier

Details A dummy DMA RX request is generated as soon as the SPI is configured in *slave* mode and a SPI clock edge is detected. The dummy DMA RX request is generated during module configuration, when the module is not performing an SPI transfer. The dummy DMA RX request occurs as soon as the SPI interface signal sensitivity is changed compared to the default value.

The dummy DMA RX request is generated because the mechanism to avoid dummy data capture on a CS glitch is done regardless of channel activation.

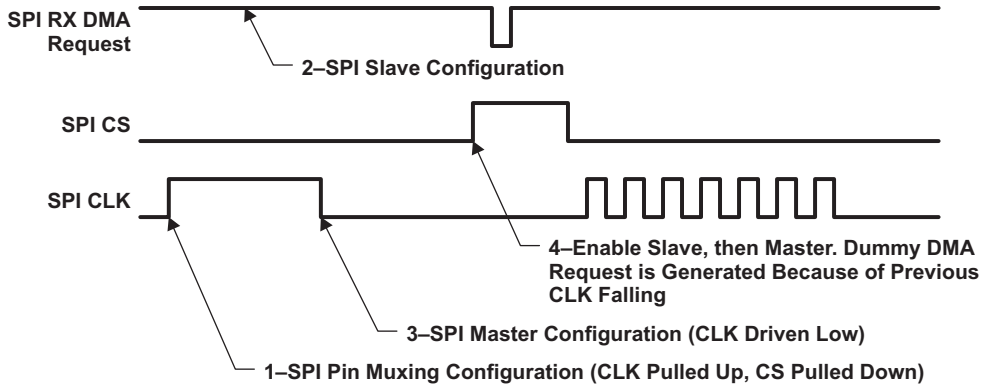


Figure 9. SPI Dummy DMA RX Generation

Workaround(s) Avoid conditions where the SPI is in *slave* mode and the SPI clock toggles (see examples below).

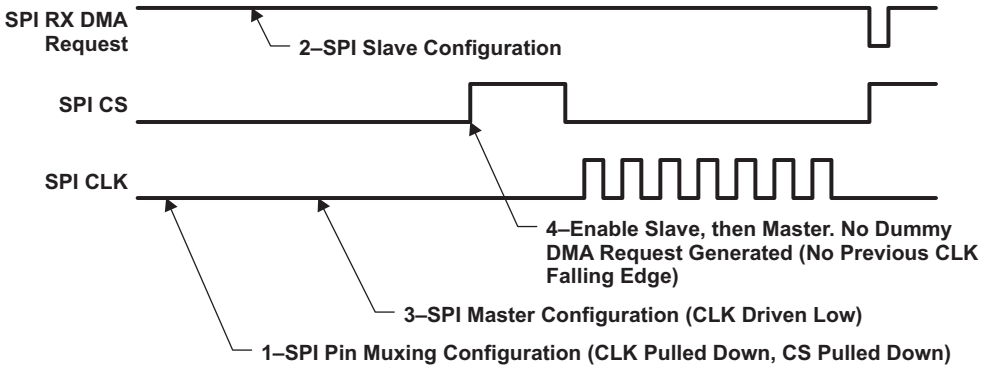


Figure 10. Dummy DMA RX Generation (No Clock Edge)

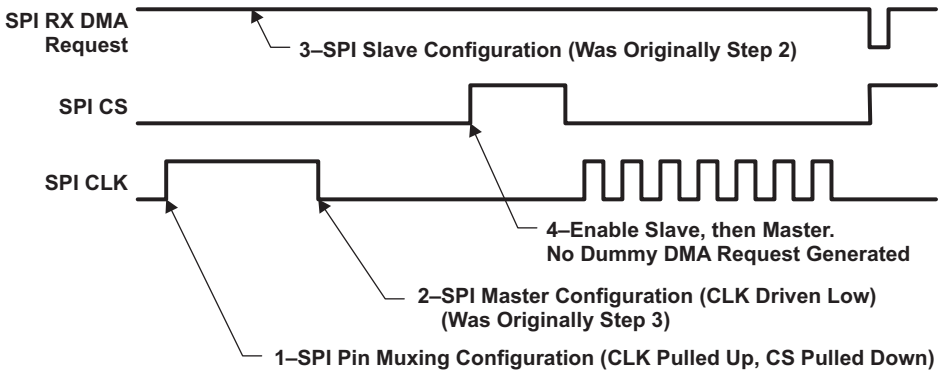


Figure 11. Dummy DMA RX Generation (Slave Mode After Clock Line Driven to Default Value)

Advisory 3.0.1.66 *SDMA: DMA4_IRQSTATUS_Lx and DMA4_IRQENABLE_Lx Registers are not Secure*

Revision(s) Affected 3.0 and earlier

Details The DMA4_IRQSTATUS_Lx and DMA4_IRQENABLE_Lx registers, where x is 0, 1, 2 or 3 are not protected in secure and supervisor modes. Therefore, the channel interrupt line of a secure / supervisor channel can be asserted. However, it is not possible to set or reset an interrupt event.

Workaround(s) No workaround.

Advisory 3.0.1.74 *PRM_VOLTCTRL and PRM_CLKSRC_CTRL Registers Reset on Warm Reset*

Revision(s) Affected 3.0 and earlier

Details The PRM_VOLTCTRL and PRM_CLKSRC_CTRL registers are reset on a Warm Reset; however, they should be reset on Cold Reset only. These parameters depend on the device environment only, but the registers must be re-programmed.

Workaround(s) No consequence. If default values are used, the registers must be re-programmed at Warm Reset release.

Advisory 3.0.1.75 *IVA2: CAM/SGX Dependencies (OMAP3530/25 only)*

Revision(s) Affected 3.0 and earlier

Details Scenario:

- IVA2 is idled.
- Interrupt is propagated to the IVA2.
- IVA2 INTC (WUGEN) generates a Wake-Up event to the PRCM for IVA2 Wake-Up.
- A Wake-Up dependency is defined between IVA and CAM, or IVA and SGX.
- PRCM wakes-up IVA2, which initiates wake-up for CAM/SGX.
- Interrupt is propagated to IVA2 Core.
- At this point IVA2 Core could initiate a transfer to CAM/SGX even though the CAM/SGX module may not have finished its wake-up sequence, thus resulting in OCP transfer fail.

Workaround(s) The IVA2 software should look at the CAM/SGX clock activity status bit to verify that the CAM/SGX domain is ON before performing any accesses. Next, 10 NOPs should be inserted for additional margin.

Advisory 3.0.1.76 *Erroneous SResp Generation Issued to AES Immediately After Soft Reset*

Revision(s) Affected	3.0 and earlier
Details	An OCP bus error (SRESP) occurs when an access to the module is performed while the module is coming out of Soft Reset.
Workaround(s)	Insert 2 NOPs after Soft Reset assertion.

Advisory 3.0.1.77 *MPU L2 Cache Size Status Register Value Inverted*

Revision(s) Affected 3.0 and earlier

Details The MPU L2 Cache Size Status register value is inverted compared to the value given in the spec. The MPU L2 cache size status (CONTROL_FEATURE_OMAP_STATUS[11:10].MPU_L2_CACHESIZE) is inverted:

Expected:

- 00 = 0KB
- 01 = 64KB
- 10 = 128KB
- 11 = 256KB

Current implementation:

- 00 = 256KB
- 01 = 128KB
- 10 = 64KB
- 11 = 0KB

Workaround(s) The software should appropriately handle the inversion.

Advisory 3.0.1.81 *L3 Interconnect Clock Divisor Default Value Must be Modified Before Configuration of SDRAM Controller*

Revision(s) Affected

3.0 and earlier

Details

The SDRAM output clock is gated when an incorrect L3 CLK_SEL ratio is set. This operation is generally transparent and handled at boot time by the ROM code. The workaround should be implemented for the following scenarios:

- GP device External Fast boot is used.
- EMU device External boot is used.

Workaround(s)

For External Boot or External Fast Boot which are both impacted by the issue, CLKSEL_L3 (bit 1:0) = 10b must be set before performing SDRC configuration. In all other cases, this programing is handled by the ROM code and no workaround is necessary.

Advisory 3.0.1.112 *DMA: Drain_IE Reset Value*

Revision(s) Affected 3.0 and earlier

Details The Drain_IE bit in the DMA4_CICRi[12] register is not initialized after POR, and its default value can be either '0' or '1' while the documentation specifies '0'.

Workaround(s) Prior to the DMA setup, the software must write 0x0 to this bit to disable the Drain_IE interrupt in the CICR register.

Advisory 3.0.1.113 *sDMA: Channel is Not Disabled After a Transaction Error*

Revision(s) Affected 3.0 and earlier

Details During a destination synchronized transfer on the write port (or source sync with SDMA.DMA4_CCRin[25] BUFFERING_DISABLE = 1), if a transaction error is reported at the last element of the transaction, the channel is not automatically disabled by DMA.

Workaround(s) Whenever a transaction error is detected on the write side of the channel, the software must disable the channel by writing a '0' to DMA4_CCRi[7]: ENABLE bit.

Advisory 3.0.1.117 *Unexpected Stalling May Occur During SDMA/IDMA Accesses to DSP L2 Memory*

Revision(s) Affected 3.0 and earlier

Details

NOTE: This exception does not apply if there are no IDMA/SDMA accesses to DSP L2 memory (both cache/SRAM and shared SRAM) during run-time.

Background Information:

The C64x+ Megamodule in the IVA2.2 subsystem has a Master Direct Memory Access (MDMA) bus interface and a Slave DMA (SDMA) bus interface (see Figure 12). The MDMA interface provides DSP access to resources outside the C64x+ Megamodule such as external DDR memory, flash memory, and On-chip Memory (OCM) RAM. The MDMA interface is typically used for C64x+ CPU/cache accesses to memory beyond the Level 2 (L2) memory level. These accesses can be in the form of cache line allocates, write-backs, and non-cacheable loads and stores to/from system memories.

The SDMA interface allows other master peripherals in the system to access DSP memories. The memories which may be accessed are Level 1 Data (L1D), Level 1 Program (L1P), L2 cache/RAM, and shared L2 RAM. Examples of masters who may access these memories are EDMA transfer controllers, the system DMA, and the MPU.

Note: The IVA2.2 video hardware accelerator accesses L2 shared RAM through the shared L2 interface (SL2IF). Accesses through the shared L2 SRAM are not susceptible to this issue.

The DSP Internal DMA (IDMA) is a DMA engine within the C64x+ Megamodule which is used to move data between internal DSP memories (L1 and L2) and/or DSP peripheral configuration bus. The IDMA engine shares resources with the SDMA interface.

The C64x+ Megamodule has a L1D cache and L2 cache both implementing write-back data caches. This means that it holds updated values for external memory as long as possible. It writes these updated values to external memory either when it needs to make room for new data, or when specifically requested to do so by the application. These write-backs are called “victims.” The L1D sends its victims to L2. The caching architecture has pipelining which means multiple requests could be pending between L1, L2 and MDMA.

Additional details on the C64x+ Megamodule and its MDMA and SDMA ports can be found in the *TMS320C64X+ Megamodule Reference Guide* (literature number SPRU871).

Exception Description:

Ideally the MDMA and SDMA/IDMA paths operate independently with minimal interference. Normally MDMA accesses may stall for extended periods of time (clock cycles) due to expected system level delays (e.g. due to bandwidth limitations and/or DDR memory refreshes). However, due to the exception, there are cases where SDMA and/or IDMA accesses to L2/L1 experience an additional/unexpected stall during the normal stalls seen by the MDMA interface. For latency sensitive traffic, the SDMA stall can result in missing real time deadlines.

NOTE: SDMA/IDMA accesses to L1P/D will not see any unexpected stall if there are no SDMA/IDMA accesses to L2 (both cache/SRAM and shared SRAM). This is because unexpected SDMA/IDMA stalls to L1 happen only when they are pipelined behind L2 accesses which experience stalls due to the exception.

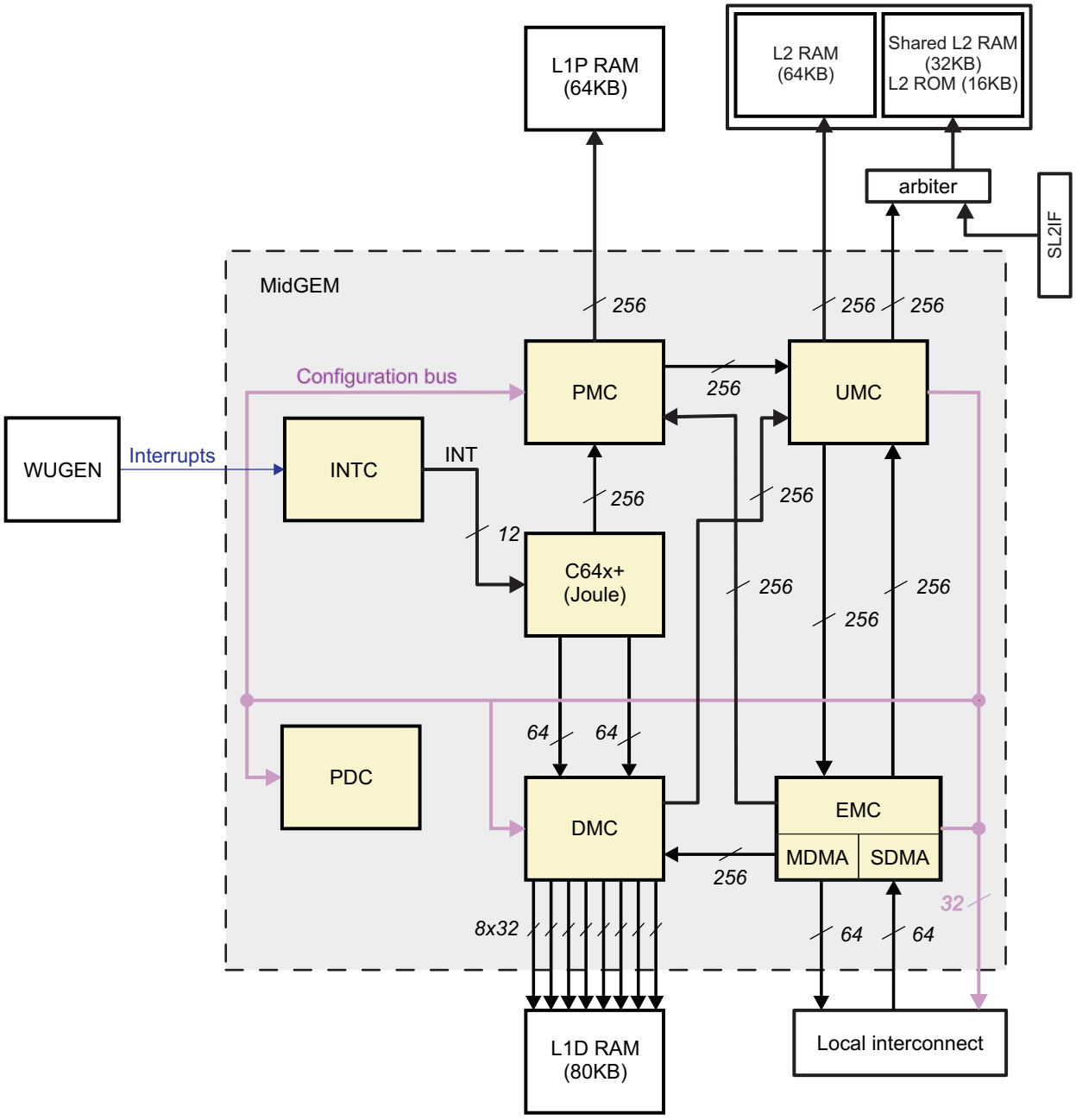


Figure 12. C64x+ Megamodule Block Diagram

In the following scenarios, SDMA/IDMA stalls may possibly occur. Note that each of these scenarios describes expected normal DSP functionality. The only issue is that the SDMA/IDMA access potentially exhibits additional unexpected stalling during each of these scenarios:

1. Bursts of writes to non-cacheable MDMA space (external DDR memory, flash memory, and OCM RAM). The DSP buffers up to 4 non-cacheable writes. When this buffer fills, SDMA/IDMA is blocked until the buffer is non-full. Thus, bursts of non-cacheable writes longer than three writes can stall SDMA/IDMA traffic.
2. Various combinations of L1 & L2 cache activity:
   a. L1D read miss generating victim traffic to L2 (Cache or SRAM) or external memory. The SDMA/IDMA may be stalled while servicing the read miss and the victim. If the read miss also misses L2 cache, the SDMA/IDMA may be stalled

until data is fetched from external memory to service the read miss.

b. L1D read request missing L2 (going external) while another L1D request becomes pending. The SDMA/IDMA may be stalled until the external memory access is complete.

c. L2 victim traffic to external memory during any pending L1D request. SDMA/IDMA may be stalled until external memory access and the pending L1D request completes.

The duration of the IDMA/SDMA stalls is highly dependent on the quantity/characteristics of the L1/L2 cache traffic and MDMA traffic present in the specific system/application and is not simple to quantify. In case # 2a, 2b and 2c from above, stalling may or may not occur depending on specific state of the cache request pipelines and target locations of the traffics. These stalling mechanisms may also interact in various ways to cause combined stalls of larger duration. Due to these factors, it is not straightforward to predict if the stalling will occur and the duration of the stalling.

The occurrence of such IDMA/SDMA stalling and/or any system impact of such stalling will be most likely in systems with very excessive context switching and/or L1/L2 cache miss/victim traffic and in systems with very heavily loaded EMIF, again making it difficult to predict occurrence and precise duration of individual stalls.

Determining if a real-time deadline miss is due to this exception:

The steps outlined below can be used to determine if the issue described here is the culprit of real-time deadline misses for existing applications. Examples of situations in which real-time deadlines may be being missed include loss of McBSP samples and low peripheral throughput.

Step 1:

Determine if the transfer missing the real-time deadline is directly accessing L2 or L1D memory. If not, then this issue is not the source of the problem.

Step 2:

Next, identify all SDMA transfers to/from L2 memory. An EDMA transfer to/from L2 from/to a McBSP would be an example of such a transfer. Another example would be an system DMA block transfer to/from L2.

If there are no SDMA transfers going into L2, then this issue is not the source of the problem.

Step 3:

Redirect all SDMA transfers to L2 memory to other memories. This can be done several ways:

- Temporarily move all the L2 SDMA transfers to L1D SRAM.
- If not all L2 SDMA transfers can be moved to L1D memory, temporarily direct some of the transfers to OCM RAM or DDR memory and keep the rest in L1D memory. Still, there should be no L2 SDMA transfers.

If real-time deadlines are still being missed after implementing any of the options in Step 3, then the issue described here is likely not the cause of the problem. If real-time deadline misses are solved using any of the options in Step 3, then it is very likely this issue is the source of the problem. Refer to “Workaround Description” for ways to work around this issue.

Workaround(s)

Method 1

Entirely eliminate the exception by removing all SDMA/IDMA accesses to L2 SRAM. The EDMA/QDMA, IDMA, system DMA, and other chip masters cannot access L2. There are no issues with the CPU itself accessing code/data in L2. Also, this issue only pertains to SDMA accesses to L2, accesses through the shared L2 interface (SL2IF) are still allowed. Note that it is recommended to always configure the L2 cache/SRAM block as 100% cache.

Method 2

Issues such as dropped McBSP samples can be worked around by moving latency sensitive buffers outside the C64x+ Megamodule. For example, rather than placing buffers for the McBSP into L1/L2, those buffers can instead be placed in other memory such as OCM RAM.

Method 3

Employ any combination of the following as appropriate/possible to reduce the system impact of the exception:

1. Improve system tolerance on DMA side (IDMA/SDMA/MDMA):
   a. Understand and minimize latency critical SDMA/IDMA accesses to L2 or L1P/D.
   b. Directly reduce critical real-time deadlines if possible at peripheral/IO level (e.g. increase word size and/or reduce bit rates on serial ports).
   c. Reduce DSP MDMA latency by:
      i. Increase priority of DSP access to external memory such that MDMA latency of MDMA accesses causing stalls is minimized (NOTE: other masters such as system DMA may have real-time deadlines which dictate higher priority than DSP).
      ii. Do not perform external memory access using ready handshaking during DSP run-time (devices using ready handshaking potentially insert very excessive latency to external memory accesses).
2. Minimize offending scenarios on DSP/Caching side:
   a. If DSP performing non-cacheable writes is causing issue, insert non-cacheable reads every few writes to allow write buffer to drain.
   b. Avoid caching from slow memories such as asynchronous memory. Instead, page the data via the EDMA from the off-chip async memory to L2 SRAM or SDRAM space before accessing the data from the DSP. NOTE: paging cannot occur while real-time deadlines must be met.
   c. Use explicit cache commands to trigger cache write-backs during appropriate times (L1D Writeback All, L2 Writeback All). Conversely, do not use these commands at inappropriate times (when real-time deadlines must be met).
   d. Restructure program data and data flow to minimize the offending cache activity
      i. Define the read-only data as “const". The const C keyword tells the compiler that the array will not be written to. By default, such arrays get allocated to the “.const” section as opposed to BSS. With a suitable linker command file, the developer can link the .const section off-chip, while linking .bss on-chip. Because programs initialize .bss at run time, this has the added benefit of reducing the program’s initialization time and total memory image.
      ii. Explicitly allocate lookup tables and writeable buffers to their own sections. The #pragma DATA_SECTION(label, “section”) directive tells the compiler to place a particular variable in the specified COFF section. The developer can explicitly control the layout of the program with this directive and an appropriate linker command file.
      iii. Avoid directly accessing data in slow memories (e.g. flash), copy at init time to faster memories
   e. Modify troublesome code
      i. Rewrite to use DMAs to minimize data cache writebacks. If the code accesses a large quantity of data externally, consider using DMAs to bring in the data, using double buffering and related techniques (minimizing cache write-back traffic and thus chance for occurrence of the exception).
      ii. Re-block the loops. In some cases, restructuring loops can increase reuse in the cache, and reduce total traffic to external memory.
      iii. Throttle the loops. If restructuring the code is impractical, then it becomes reasonable to slow it down. What this does is reduce the likelihood that consecutive SDMA/IDMA blocks “stack up” in the cache request pipelines resulting in a very long stall.

Advisory 3.0.1.119 *USB DEVICE Aborts Remote Wakeup Sequence When OMAP35x Device Wakes From OFF/RET to ON in USB TLL Mode*

Revision(s) Affected

3.0 and earlier

Details

When the OMAP35x device is programmed to go to OFF/RET mode, the power, reset, and clock management module (PRCM) powers down the High-speed USB Host Subsystem. In response, the High-speed USB Host Controller suspends the bus and the USBTLL issues a suspend interrupt.

Before USB Host Controller is switched OFF its register contents are automatically saved to voltage domain retention memory. Also, before the CORE voltage domain is switched to OFF/RET, the USBTLL contents are saved. Once these registers are saved, the OMAP35x device transitions to the OFF/RET state.

When an external device initiates a remote wakeup, the PRCM wakes the OMAP35x device. The CORE domain reset is released and the USBTLL registers are restored. Upon the completion of the USBTLL register restore an ALT interrupt is erroneously generated by the USBTLL to the external device. This breaks the USB remote wakeup protocol and as a result the external device aborts the remote wakeup sequence.

Workaround(s)

To workaround this issue the IdGnd fall interrupts should be disabled by the external device at boot-up. This interrupt can be disabled by clearing the IDGND_FALL bit of the ULPI_USB_INT_EN_FALL_i registers.

Advisory 3.0.1.127 *Pending EDMA Interrupts in Video Hardware Accelerator Prevent IVA2 From Going Into Idle Mode*

Revision(s) Affected

3.0 and earlier

Details

The video hardware accelerator module included inside IVA2.2 can be placed in reset through the Power, Reset, and Clock Management (PRCM) module through the IVA2_RST3 signal.

However, if the video hardware accelerator is placed in reset, and the EDMA generates an interrupt to the video hardware accelerator , the interrupt will not be acknowledged. The pending interrupt condition will prevent IVA2.2 from going into idle mode.

Workaround(s)

There are several workarounds for this issue.

Workaround A: If the video hardware accelerator is not being used, then this issue does not apply. However, pending interrupts can always be cleared from the video hardware accelerator by toggling IVA2_RST3.

Workaround B (TBD): If the video hardware accelerator is used, then call the following software function to clear this error condition: <TBD>.

Advisory 3.0.1.129 *TV detect AC coupling Mode Not Supported*

Revision(s) Affected	3.0 and earlier
Details	The TV detect in AC coupling mode is not implemented accurately and is not functional; therefore, TV detection in AC mode is impossible.
Workaround(s)	Use DC coupling mode only.

Advisory 3.0.1.130 *USB DMA Cannot Handle Concurrent Channels*

Revision(s) Affected 3.0 and earlier

Details USB DMA can be used only with one channel (Rx or Tx) active at a time. When more than one channel is active, then DMA transfers cannot be guaranteed.

Workaround(s) **Workaround A**: Use the interrupt mode.

Workaround B: Use Tx DMA mode1 for highest throughput requirement endpoint. Use interrupt mode for others.

Workaround C: Use Rx DMA mode0 for highest throughput requirement endpoint and interrupt mode for others.

Advisory 3.0.1.131 *VC1 En/De-coded Bit Stream Corrupted When iLF is Used*

Revision(s) Affected 3.0 and earlier

Details Due to a hardware issue in an arithmetic operator in the iLF accelerator, the output of the loop filter algorithm is corrupted (truncation error). This operator is not used in any other block of the IVA2.2 subsystem. Other multimedia codecs (H264, RV9, WMV9, …) using iLF are not impacted by this bug. It is safe to use iLF for codecs other than VC1.

Note: This bug does not create a noticeable impact on the video quality. Even if there is no visible impact on the video quality for the end-user, using iLF loop filter for VC1 encoding or decoding generates a video stream that is not bit exact as compared to the reference: *SMPTE Standard for Television: Compressed Video Bitstream Format and Decoding Process*.

Workaround(s) iLF accelerator cannot be used for VC1 decoding and will not produce a bit exact output compliant with SMPTE standard. A SMPTE compliant bit stream can be achieved by using IVA2.2 software solution (iLF not used ,and loop filtering handled by DSP software).

Advisory 3.0.1.136 *ISP: LSC Issue When Used Concurrently With Resizer*

Revision(s) Affected 3.0 and earlier

Details Due to faulty arbitration in the SBL (shared buffered logic: local interconnect of the ISP), some LSC coefficients can be incorrectly applied on the image. This issue appears only if the preview resizer is accessing the SBL simultanously with LSC table prefetch. The issue appears randomly depending on the access sequencing between resizer access and LSC accesses. The occurrence of the issue increases with larger image size and smaller paxel size. The corrupted LSC coefficients are not random values, but value of a neighbor (the artifact is difficult to detect with regular values but can be clearly seen with a dedicated LSC pattern).

Workaround(s) The workarounds consist of avoiding concurrent accesses to the SBL from the resizer and LSC which can be implemented using various methods. TI is currently validating the different options, and this section will be updated once the optimal workaround in terms of efficiency and software complexity has been confirmed.

Advisory 3.0.1.137 *DALL Leakage When Placed in Low-Power Stop Mode*

Revision(s) Affected 3.0

Details Due to a DALL integration issue, leakage is detected on VDDS_DPLL_DLL (MPU/IVA) and VDDS_DPLL_PER (per DPLLs) rails when DALL is placed in low-power stop mode. This issue is due to DALL going to fast relock mode instead of going to low-power stop mode.

Notes:

- Device OFF mode and CORE DALL are not impacted.

Workaround(s) There is currently no workaround in place, but under evaluation, this section will be updated.

Advisory 3.0.1.140 *ISP Lens Shading Correction Issue*

Revision(s) Affected

3.0 and earlier

Details

The Lens Shading Correction (LSC) module in CCDC cannot be used simultaneously with the Preview-to-Resizer path, as simultaneous accesses can result in data corruption.

The LSC data is randomly dropped, resulting in a color-shifted final image. This issue only occurs when LSC is used simultaneously with Preview-to-Resizer.

Workaround(s)

TI is investigating a work-around in which the LSC in CCDC is used along with the Resizer in mem-2-mem operation.

Other possible work-around options for OEM/3P imaging software are possible:

- Use the LSC available in the Preview module.
- Modify the sensor timings to adjust the Resizer/CCDC timing so that the issue does not occur.

3 Silicon Revision 2.1 Usage Notes and Known Design Exceptions to Functional Specifications

This section describes the usage notes and advisories that apply to silicon revision 2.0 of the OMAP35x Applications Processor.

Note: The peripherals supported on the various OMAP35x Applications Processor devices are different. The user should only refer to usage notes and advisories pertaining to features supported on the specific device. For a complete list of the supported features of the OMAP35x Applications Processor, see the device-specific data manuals.

3.1 Usage Notes for Silicon Revision 2.1

Silicon revision 2.1 applicable usage notes have been found on a later silicon revision. For more details, see Section 2.1, *Usage Notes for Silicon Revision 3.0*.

3.2 Silicon Revision 2.1 Known Design Exceptions to Functional Specifications

Some silicon revision 2.1 applicable advisories have been found on a later silicon revision. For more details, see Section 2.2, *Silicon Revision 3.0 Known Design Exceptions to Functional Specifications*.

Table 6. Silicon Revision 2.1 Advisory List

Title	Page
Advisory 2.1.1.101 Dummy Data Sent When Enabling the Video Mode	53
Advisory 2.1.1.120 SYS_CLKOUT1 Not Enabled by ROM Code	54
Advisory 2.1.1.122 ROM code: Incomplete ASIC ID	55
Advisory 2.1.1.123 ROM Code: Context Restore Failure if OCM RAM is OFF After Wakeup	56
Advisory 2.1.1.128 MMC: Multiple Block Read Operation Issue	57
Advisory 2.1.1.135 Isolation Issue Between SIM_VDDS and VDDS	58
Advisory 2.1.1.141 CPU Revision Code Register Does Not Return Same Value for ES2.0 and ES2.1 Samples	59

Advisory 2.1.1.101 *Dummy Data Sent When Enabling the Video Mode*

Revision(s) Affected 2.1 and earlier

Details Under specific conditions described below, some dummy data of a transfer aborted in *video* mode are kept in the line buffer and would be sent inaccurately when *video* mode is re-enabled. The following conditions will cause this issue to occur:

1. Only one blanking line before first active line in the frame which means: VSA = 1, VBP = 0
2. *Video* mode is disabled during the first line (VSA line) on VP.
3. Small HBP (i.e., less than 40)

Note: This scenario is not expected in real applications. The issue occurrence increases with smaller lines, although the issue can occur with any line size.

Workaround(s) If conditions 1, 2 and 3 are met, then the *video* mode should not be disabled during the first line.

Advisory 2.1.1.120 *SYS_CLKOUT1 Not Enabled by ROM Code*

Revision(s) Affected 2.1 and earlier

Details **Note**: This issue only applies in cases where the sys_clkout1 pin of the OMAP35x device is required during boot.

As described in the *OMAP35x Technical Reference Manual (literature number* SPRUF98*)*, the sys_clkout1 pin can be used to output a clock which can be used by other devices in the system. However, before being used, the sys_clkout1 pin configuration must be changed from its default “safe mode” configuration to its “sys_clkout1” configuration.

During boot, the ROM code does not change the sys_clkout1 configuration to “sys_clkout1”. Therefore, the sys_clkout1 cannot be enabled until the boot process is complete.

For correct operation, the ROM code must set the sys_clkout1 pin for “sys_clkout1” configuration when the local system clock crystal oscillator is used (sys_boot6 = 0). However, the pin sys_boot6 is not accessible through a memory mapped register and therefore the ROM code cannot change the pad configuration according to the crystal mode.

Workaround(s) No workaround.

Advisory 2.1.1.122 *ROM code: Incomplete ASIC ID*

Revision(s) Affected 2.1 and earlier

Details The length of the root key hash has been extended on the OMAP35x architecture. During peripheral booting the ROM Code sends out an ASIC ID which - among other information - contains also the root key hash. The length of this hash field in the ASIC ID is correctly set to 20bytes but the 4 last bytes are all zero.

There is no known limitation as the root key hash field inside the ASIC ID is not necessary for peripheral booting.

Workaround(s) There is no workaround for this issue.

Advisory 2.1.1.123 *ROM Code: Context Restore Failure if OCM RAM is OFF After Wakeup*

Revision(s) Affected 2.1 and earlier

Details The context restore procedure will fail if the On-Chip Memory (OCM) RAM is OFF after wakeup.

The OCM RAM will be off after wakeup if bits MEM1ONSTATE and MEM2ONSTATE in the PM_PWRSTCTRL_CORE register are set to OFF (0x0) before going to sleep.

Workaround(s) The application software must make sure that the bits MEM1ONSTATE and MEM2ONSTATE are set to ON (0x3) or RETENTION (0x1) before putting the CORE domain to sleep. There is no need to set the MEMORYCHANGE bit in the same register.

Advisory 2.1.1.128 *MMC: Multiple Block Read Operation Issue*

Revision(s) Affected 2.1 and earlier

Details The multiple block read transfer, in polling and interrupt mode, does not work correctly on MMC1 and MMC2. A Data CRC error is generated due to some corrupted data, when the read buffer (two 512-bytes portions) of the MMC controller is full.

If the buffer is not free, the MMC controller stops the clock and the card stops to send data to . The clock will be re-enabled only when one portion will be emptied and then the transfer will restart. The output clock and data enable are generated on same internal clock edge. The data enable signal must be re-asserted before the first clock edge of the input clock (feedback clock) in order to sample the data correctly.

Hold buffers have been added on data enable signal that makes data enable arrival after the first clock edge. It leads that the first data is not sampled on clock restart.

This issue depends on the pattern written and the output data width (1-, 4- or 8-bits mode). For example, Pattern written: 0x2800, Pattern read: 0x0801.

No data are read by the MPU into the read buffer in order to fill the 2 portions. The data enable signal is de-asserted and the output clock is disabled.

- In 1-bit mode, the failure occurs on the second data of third block when the MPU starts to read the data.
- In 4-bits mode, the failure occurs on the third data of third block when the MPU starts to read the data.

This issue is related to the integration of the IP and is not a functional bug.Therefore, MMC3 is not impacted.

Workaround(s) There is no workaround for this issue.

Advisory 2.1.1.135 *Isolation Issue Between SIM_VDDS and VDDS*

Revision(s) Affected	2.1 and earlier
Details	There is no functional impact, but there is leakage current of approximately 20mA from SIM_VDDS to VDDS due to an isolation issue.
Workaround(s)	There is no workaround for this issue.

Advisory 2.1.1.141 *CPU Revision Code Register Does Not Return Same Value for ES2.0 and ES2.1 Samples*

Revision(s) Affected

2.1

Details

The CPU revision code register should return the same value for both ES2.0 and ES2.1 samples. However, the version of the cortexA8 core is incorrectly read. The results of the read are shown in Table 7.

Table 7. CPU Revision Code Register Readings

	EXPECTED READING	ACTUAL READING
ES2.0 Silicon ID	0x411fc081- r1p1	0x411fc081- r1p1
ES2.1 Silicon ID	0x411fc081- r1p1 + bug fixes	0x411fc082 - r1p2

Workaround(s)

There is no workaround for this issue.

4 Silicon Revision 2.0 Usage Notes and Known Design Exceptions to Functional Specifications

This section describes the usage notes and advisories that apply to silicon revision 2.0 of the OMAP35x Applications Processor.

Note: The peripherals supported on the various OMAP35x Applications Processor devices are different. The user should only refer to usage notes and advisories pertaining to features supported on the specific device. For a complete list of the supported features of the OMAP35x Applications Processor, see the device-specific data manuals.

4.1 Usage Notes for Silicon Revision 2.0

Silicon revision 2.0 applicable usage notes have been found on a later silicon revision. For more details, see Section 2.1, *Usage Notes for Silicon Revision 2.1*.

4.2 Silicon Revision 2.0 Known Design Exceptions to Functional Specifications

Some silicon revision 2.0 applicable advisories have been found on a later silicon revision. For more details, see Section 2.2, *Silicon Revision 2.1 Known Design Exceptions to Functional Specifications*.

Table 8. Silicon Revision 2.0 Advisory List

Title	Page
Advisory 2.0.1.52 VENC: DAC1 and DAC2 Inversion	61
Advisory 2.0.1.80 Device Stalls After a Warm Reset	63
Advisory 2.0.1.82 SDI: Incorrect Control of SDI Analog Complex IO	64
Advisory 2.0.1.104 ROM Code: MPU is Stuck in Secure Mode After Warm Reset on GP Device	65
Advisory 2.0.1.105 ROM Code: Second Power-On Reset Required for USB Boot	66
Advisory 2.0.1.106 USB: Peripheral Booting Dependent Upon System Clock Frequency	67
Advisory 2.0.1.108 ROM Code: OCM RAM is Not Turned ON Correctly After Wakeup	68
Advisory 2.0.1.109 ROM Code: Semaphore is Not Initialized in Scratchpad Memory	69
Advisory 2.0.1.110 ROM Code: MMC Booting Using 4 /8-Bit Bus may Fail	70
Advisory 2.0.1.114 Cortex-A8 r1p1 Exhibits Higher Level of Missed Branch Prediction	71

Advisory 2.0.1.52 *VENC: DAC1 and DAC2 Inversion*

Revision(s) Affected

2.0

Details

As stated in the Display Subsystem chapter of the *OMAP35x Technical Reference Manual* (literature number SPRUFA4), the video encoder DAC1 is used for luminance/composite output and the video encoder DAC2 is used for chrominance output. The video encoder DAC1 also includes a TV detection feature. The TV detection functionality is available on the OMAP35x Applications Processor video encoder DAC1, but is not available on DAC2. Due to an integration error in the video encoder, DAC1 is used for chrominance output, and DAC2 is used for luminance output. For proper TV detection in *composite video* mode, DAC1 should have been used for luminance, and DAC2 for chrominance.

However, due to an integration error in the video encoder, DAC1 is used for chrominance output and DAC2 is used for luminance output. Proper care should be taken when connecting to external devices (see workaround).

Furthermore, for proper TV detection in composite video mode (CVBS), DAC1 should be used for luminance and DAC2 for chrominance. Therefore, as a direct consequence of implementation, the TV detect feature is not available for composite video mode. Figure 13 shows the DAC implementation.

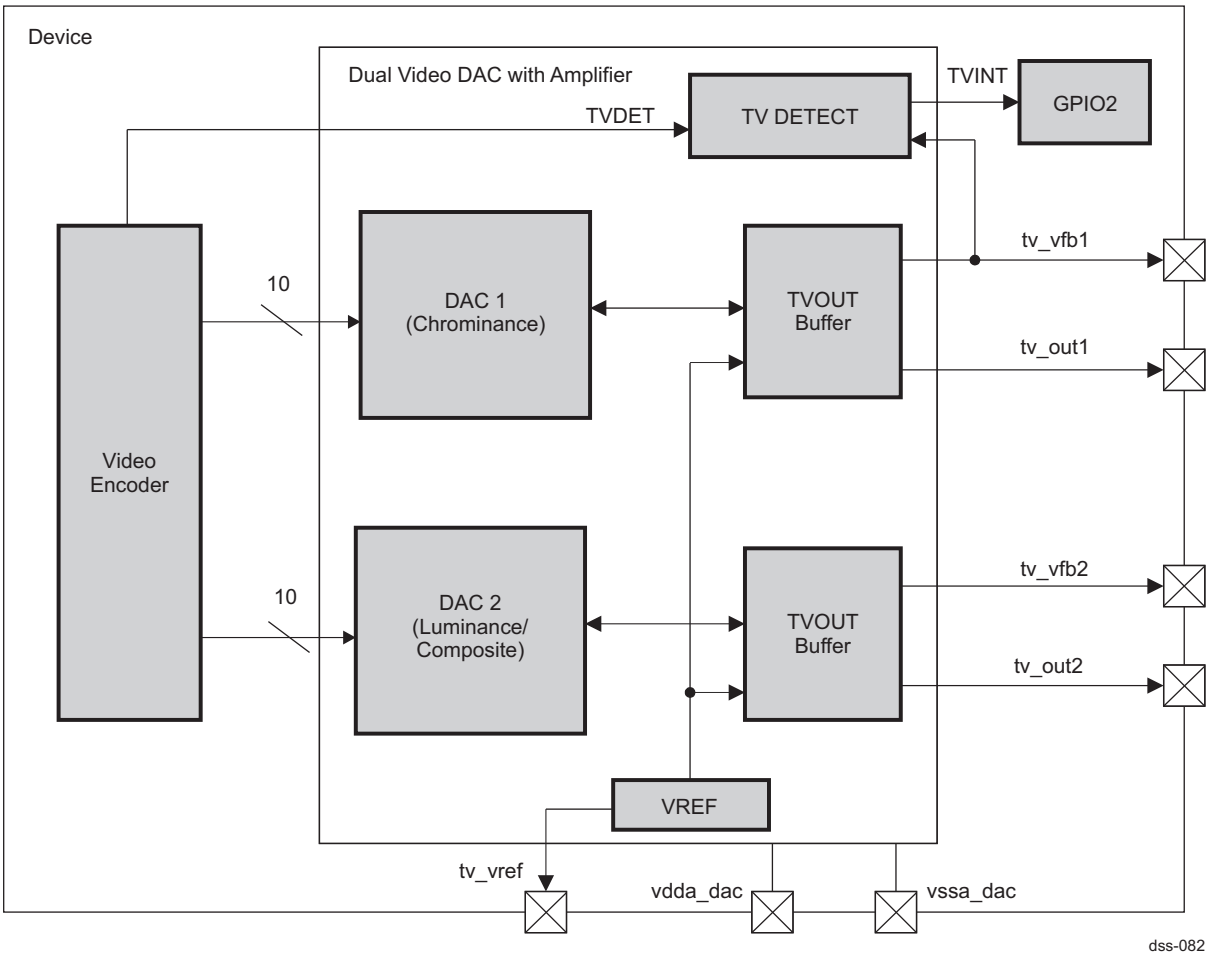


Figure 13. DAC Implementation

Workaround(s)

For proper TV out functionality, the external connection to the TV set must follow the implementation shown in Figure 14.

There is no workaround for the TV detect functionality. This feature is unavailable in *composite* video mode.

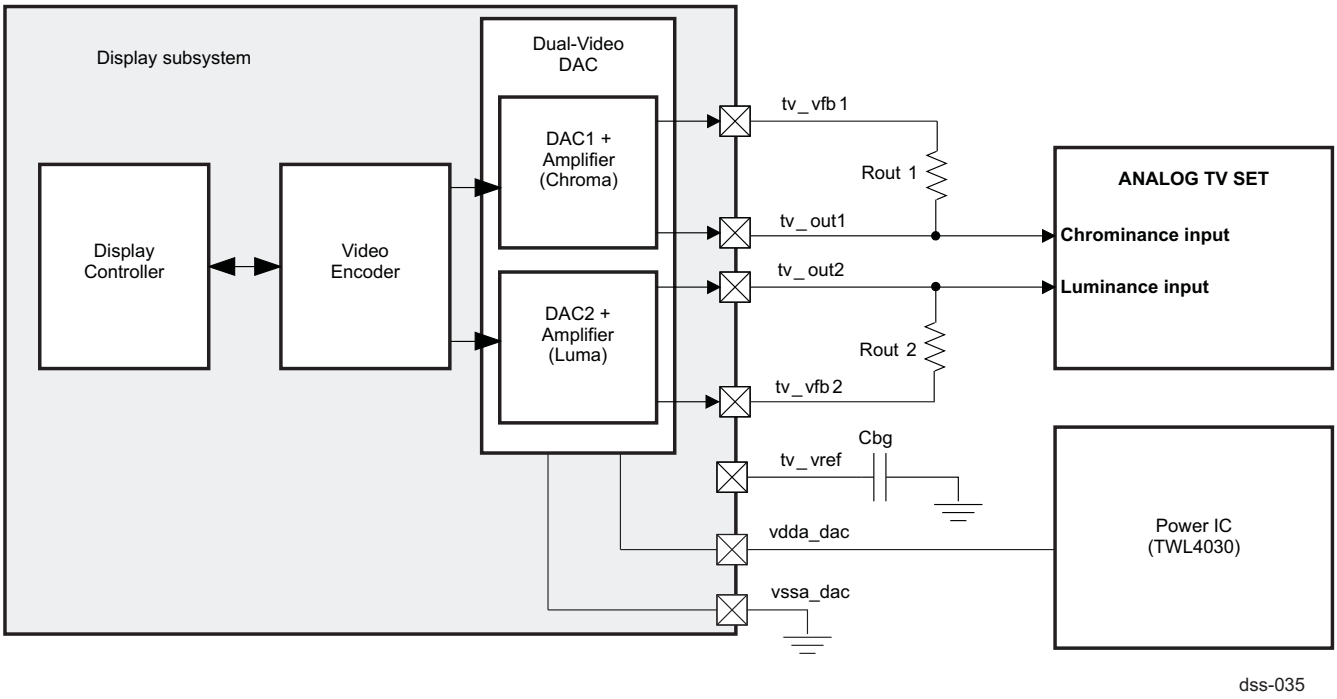


Figure 14. External Connection to TV Set

Advisory 2.0.1.80 *Device Stalls After a Warm Reset*

Revision(s) Affected 2.0

Details The PRCM fails to release the MPU reset after a number of Global Warm Reset iterations. This failure is random and can occur on any device type at any SYS_CLK frequency (i.e., 12 MHz, 19.2 MHz, 26 MHz, etc). However, the issue only occurs when DPLL3 (CORE DPLL) and DPLL4 (PER DPLL) are locked prior to generating a global software warm reset. This issue is not seen when DPLL3 is configured in *bypass* or *locked* mode and DPLL4 is in *low-power stop* mode before the Global Warm Reset has been applied.

Note: This issue affects all sources of Global Warm Reset (i.e,. global software warm reset, watchdog timer reset, external global warm reset, IcePick warm reset, etc.).

Workaround(s) Avoid Global software Warm Reset assertion or any source of Global Warm Reset while both DPLL3 and DPLL4 are locked.

Advisory 2.0.1.82 *SDI: Incorrect Control of SDI Analog Complex IO*

Revision(s) Affected 2.0

Details There is an incorrect signal connection to control the SDIv2 analog complex IO. As a result, *FlatLink3G* mode is not functional. Other serial modes (CMADS 1000, 2000) are not impacted.

Workaround(s) There is no workaround for this issue.

Advisory 2.0.1.104 *ROM Code: MPU is Stuck in Secure Mode After Warm Reset on GP Device*

Revision(s) Affected 2.0

Details After any Warm Reset, the chip will branch to secure ROM Code. Since the secure ROM code does not exist on GP devices, the MPU will remain stuck in a loop. No debug can be performed.

The ROM code does not correctly manage the different resets. For a Warm Reset, the ROM code interprets the reset as a wake-up reset. The ROM code then tries to proceed to the context restore, interpreting data stored in scratchpad memory. If the data stored in scratchpad contains rubbish, the ROM code will attempt to restore them.

Workaround(s) The scratchpad memory from physical address 0x48002910 to 0x480029FF must be cleared by the application after a Power On Reset.

Advisory 2.0.1.105 *ROM Code: Second Power-On Reset Required for USB Boot*

Revision(s) Affected 2.0

Details The VBUS detection timeout in ROM Code is too short (1ms) compared to the T2 debouncing time (30ms) which prevents the start of USB booting after board Power-On Reset. The USB boot functions correctly on subsequent nRESPWRON since T2 holds the VBUS status.

Workaround(s) Apply a second nRESPWRON to the MPU. The feasability of this workaround depends on the capability of the hardware.

Advisory 2.0.1.106 *USB: Peripheral Booting Dependent Upon System Clock Frequency*

Revision(s) Affected	2.0
Details	When using TPS69xxx PHY, USB peripheral booting is functional only with a 26MHz system clock.
Workaround(s)	Use another PHY (ISP1504) or 26MHz clock frequency.

Advisory 2.0.1.108 *ROM Code: OCM RAM is Not Turned ON Correctly After Wakeup*

Revision(s) Affected 2.0

Details The internal RAM can be turned OFF, even when the chip is completely ON, running the normal application. After a wake-up, the ROM code needs to access internal RAM. The ROM code detects the RAM has been turned off and then tries to turn it on. This operation is not performed correctly, causing the OCM RAM not to be turned ON. The ROM code loops infinitely, waiting for the RAM to be ON.

Workaround(s) Before applying any sleep mode, the application must ensure that the OCM RAM is configured to be ON when the chip is ON.

Advisory 2.0.1.109 *ROM Code: Semaphore is Not Initialized in Scratchpad Memory*

Revision(s) Affected 2.0

Details The booting image can contain a configuration header that can enable the ROM code to configure SDRC. When trying to configure the SDRC, the ROM code grabs a semaphore that is stored in scratchpad memory, but this semaphore was not initialized at POR.

The value stored in this register may prevent the ROM code from booting or performing wake-up reset when SDRC restoration is necessary.

Workaround(s) To prevent any wake-up problem, the software must delete the semaphore content after POR. However, there is no workaround to prevent any freeze at POR if a configuration header is defined in the booting image.

Advisory 2.0.1.110 *ROM Code: MMC Booting Using 4 /8-Bit Bus may Fail*

Revision(s) Affected 2.0

Details The booting image that is stored on the MMC can contain a software booting configuration (SWBCFG) [a specific data header that is located just before the booting code]. This SWBCFG is interpreted by the ROM code. The SWBCFG contains 2 important fields: MMC clock and MMC bus width. When setting the MMC bus width to 4, or 8 bits, the ROM code may fail to boot on certain MMC or SD devices.

This failure depends on the MMC type. When switching to 4 or 8-bit bus width, the busy signal (present on DAT0 line) is asserted by the MMC. Some MMC devices need more than 100 Ms to be operational after having changed the bus width. After 60 Ms, the ROM code tries to read data from MMC without checking the busy signal level, and the read operation fails. MMC1 and MMC2 are impacted.

Workaround(s) Use the MMC in *1-bit* mode at booting phase.

Advisory 2.0.1.114 *Cortex-A8 r1p1 Exhibits Higher Level of Missed Branch Prediction*

Revision(s) Affected	2.0
Details	Due to an integration issue (test signal not tied to the right value), a higher level than expected of missed branch predictions are generated.
Workaround(s)	There is no workaround.

Appendix 1 Revision History

This silicon errata revision history highlights the technical changes made to the OMAP35x device to make it a 'B' revision for Silicon Revision 3.0.

Table A-1. *OMAP35x* Revision History

SEE	ADDITIONS/CHANGES/DELETIONS
Global	Added applicable updates for Silicon Revision 3.0.
Section 1.2	Updated/Changed Figure 1, Table 1, and Table 2.
Table 5	Added the following advisories: • Advisory 2.1.1.122 -ROM code: Incomplete ASIC ID • Advisory 2.1.1.135 - Isolation Issue Between SIM_VDDS and VDDS • Advisory 2.1.1.141 - CPU Revision Code Register Does Not Return Same Value for ES2.0 and ES2.1 Samples • Advisory 3.0.1.129 - TV Detect AC Coupling Mode Not Supported • Advisory 3.0.1.130 - USB DMA Cannot Handle Concurrent Channels • Advisory 3.0.1.131 - VC1 En/De-coded Bit Stream Corrupted When iLF is Used • Advisory 3.0.1.136 - ISP: LSC Issue When Used Concurrently With Resizer • Advisory 3.0.1.137 - DPLL Leakage When Placed in Low Power Stop Mode • Advisory 3.0.1.140 - ISP Lens Shading Correction Issue

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Clocks and Timers	www.ti.com/clocks	Digital Control	www.ti.com/digitalcontrol
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265